SECTION 4: Research & Risk Management

Q7. You discover an edge in a crypto strategy with a Sharpe of 2.0 in-sample but 0.5 out-of-sample. Explain 5 possible reasons for the drop in performance. Suggest methods to diagnose and fix them.

Proof:

When a crypto strategy shows a **Sharpe ratio of 2.0 in-sample** and **0.5 out-of-sample**, it indicates **significant overfitting or model degradation**. Below are **5 possible reasons** for the performance drop, with methods to **diagnose and fix** them:

1. Overfitting to In-Sample Data

Explanation:
The model might have learned patterns that are **specific to the training data** and not generalizable.

Diagnosis:

- Check for **high model complexity** (e.g., too many parameters/features).
- Look at **performance across multiple in-sample time periods** (is it consistent?).

Fix:

- Simplify the model (e.g., reduce features or use regularization).
- Use **cross-validation** or **walk-forward analysis**.
- Penalize complexity using AIC/BIC or regularization methods.

2. Data Snooping Bias

Explanation:
The strategy might be **tuned on historical data** using the same test data repeatedly, leading to false confidence.

Diagnosis:

- Count the number of times the same test set was used for tuning.
- Run **White's Reality Check** or **Deflated Sharpe Ratio** tests.

Fix:

- Set aside a **clean, untouched out-of-sample dataset**.
- Avoid reusing test data for parameter tuning.
- Use **nested cross-validation** or robust validation procedures.

3. Market Regime Change

Explanation:
Crypto markets are volatile. The **out-of-sample period might differ in volatility, volume, or trends**, which the model wasn't trained on.

Diagnosis:

- Compare in-sample vs out-of-sample market conditions.
- Plot key indicators like volatility, volume, and correlation structure.

Fix:

- Incorporate **regime detection models** (e.g., HMMs, volatility filters).
- Add **features** that adapt to changing conditions (e.g., volatility scaling).
- Retrain frequently or use **online learning methods**.

4. Poor Feature Stability

Explanation:
Features that were predictive in-sample may be **unstable or non-stationary** out-of-sample.

Diagnosis:

- Perform **feature importance analysis** across time.
- Use **PCA** or **correlation analysis** to assess feature drift.

Fix:

- Remove or replace unstable features.
- Use **rolling windows** to update features dynamically.
- Normalize or scale features by recent history.

5. Transaction Costs or Slippage Ignored

Explanation:
In-sample results often ignore **realistic trading costs**, but these can destroy alpha out-of-sample.

Diagnosis:

- Backtest with realistic assumptions: **spread, slippage, gas fees, etc.**
- Run the strategy using **paper trading** or **live simulations**.

Fix:

- Incorporate **transaction cost models** during backtesting.
- Optimize for **net returns** (after costs), not just gross Sharpe.
- Use **execution strategies** to minimize market impact (e.g., TWAP/VWAP).

Summary Table:

Reason	Diagnosis	Fix
Overfitting	Model complexity, cross-validation	Simplify model, regularize
Data Snooping	Test reuse, Reality Check	Use untouched test data
Regime Change	Compare vol/trend across periods	Use adaptive models
Feature Instability	Feature drift analysis	Use stable, normalized features
Ignored Costs	Add costs to backtest, simulate live runs	Model realistic slippage & fees

Q8. Describe how you would:

● Monitor a live AI strategy in production

● Detect anomalies in model output or PnL

● Perform real-time risk control actions based on volatility spikes or position drift

1. Monitor a live AI strategy in production

What it means:
Keep watching your AI model when it is running in the real market.

How to do it simply:

- Show real-time graphs of model predictions, trades, and profit/loss.
- Save every prediction and trade to a log file.
- Check that the model is getting the right data and working quickly.

Example: Use a dashboard tool like **Grafana** or **Streamlit** to show live charts of model performance.

2. Detect anomalies in model output or PnL

What it means:
Catch any strange or unexpected behavior in predictions or profits.

How to do it simply:

- Set limits. For example:
  - If daily profit drops more than ₹10,000 → alert!
  - If model suddenly gives only "buy" signals for 10 minutes → alert!

- Use simple checks:
    - Is profit going down too fast?
    - Are predictions very different from usual?

Example: Send an email or SMS when profit drops too much or data looks wrong.

3. Perform real-time risk control

What it means:
Take action quickly if the market gets risky or your trades are off.

How to do it simply:

- **Volatility check**:
    - If market prices are jumping a lot → pause or reduce trading.
- **Position drift check**:
    - If you planned to buy ₹1 lakh worth but now it's ₹2 lakh → fix it.
- **Auto stop**:
    - If your loss becomes too big → stop trading for the day.

Summary :

What to Do	Simple Action
Watch the model live	Use dashboards + logs
Catch weird predictions or losses	Set alerts for profit drops or odd behavior
Control risk in real-time	Pause or adjust trades if market too wild